

EXAM 1 C.EZEMA

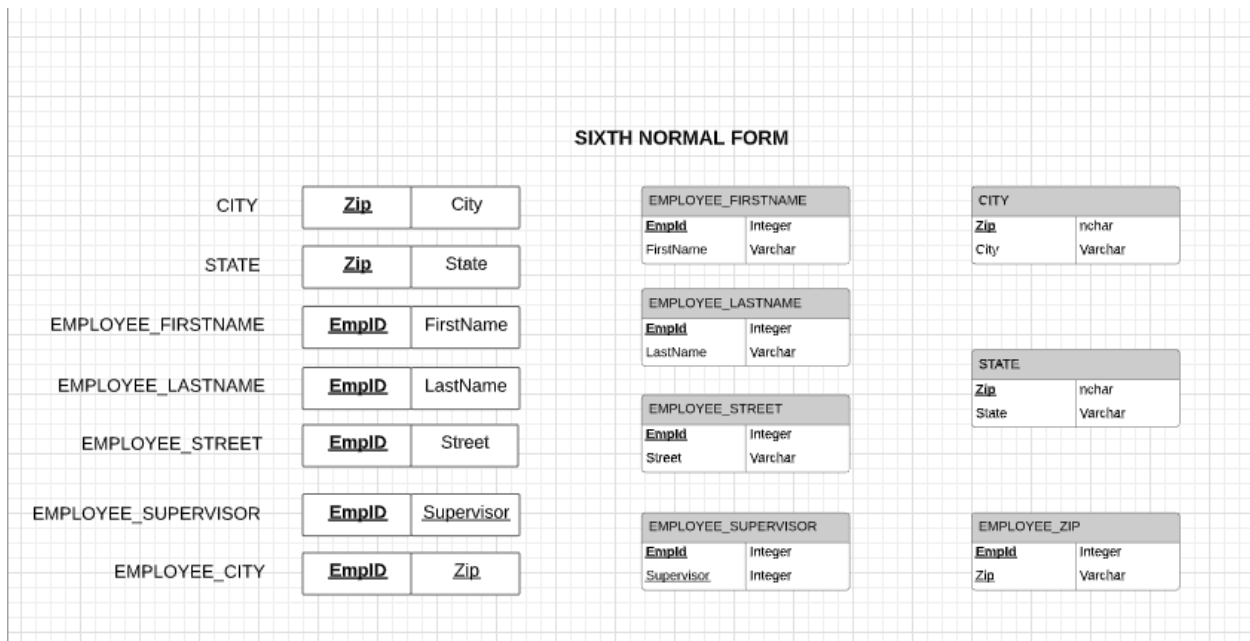
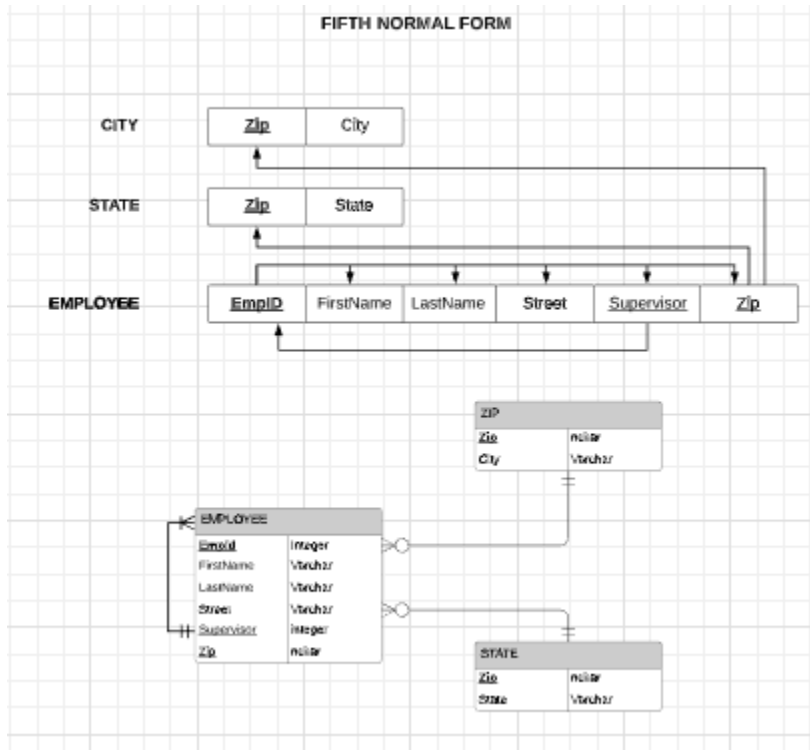
ADVANCE DATABASE

Ezema, Chukwuebuka
UNIVERSITY OF SOUTH FLORIDA

Table of Contents

Create Table Design using 5 th and 6 th Normal forms.	2
Create Database.....	3
Data Dictionary of labels, data types, constraints for Table Design.	3
Database diagram for each schema with PK/FK Relationships.....	4
Checks and Defaults	4
Rules.....	5
Triggers.....	6
Trigger 2	6
Views and SPROC's using Joins and Aggregators.....	7
USER STORIES.....	8
Left Join	8
Logic Check:.....	8
Full Join	9
Logic Check:.....	9
Right Join.....	10
Logic Check.....	10

Create Table Design using 5th and 6th Normal forms.



Create Database

Data Dictionary of labels, data types, constraints for Table Design.

C_EZEMA_EXAM1 – DATA DICTIONARY

Table: **HR.Employee**

Column Name	Description	Data Type	Size	Identity	Unique	Default	Check	Allow Nulls	Index
EmpID	PK ; Unique Employee ID	varchar	10						Y
FirstName	Employee First Name	varchar	40						
LastName	Employee Last Name	varchar	40						
DOB	Employee DOB	date					>= '1963-01-01'		
PhoneNumber	Employee Phone #	varchar	13						
Salary	Employee Salary	Money							
Email	Personal E-mail	varchar	255						
Zip	Field zip	nchar	5			33620	LIKE '[0-9][0-9] [0-9][0-9][0-9]'		

Table: **Prod.Product**

Column Name	Description	Data Type	Size	Identity	Unique	Default	Check	Allow Nulls	Index
PID	PK ; Unique Product ID	smallint							Y
Name	Product Name	varchar	14						
Type	Product Type	varchar	14			'Website'			
Price	Product Price	money							

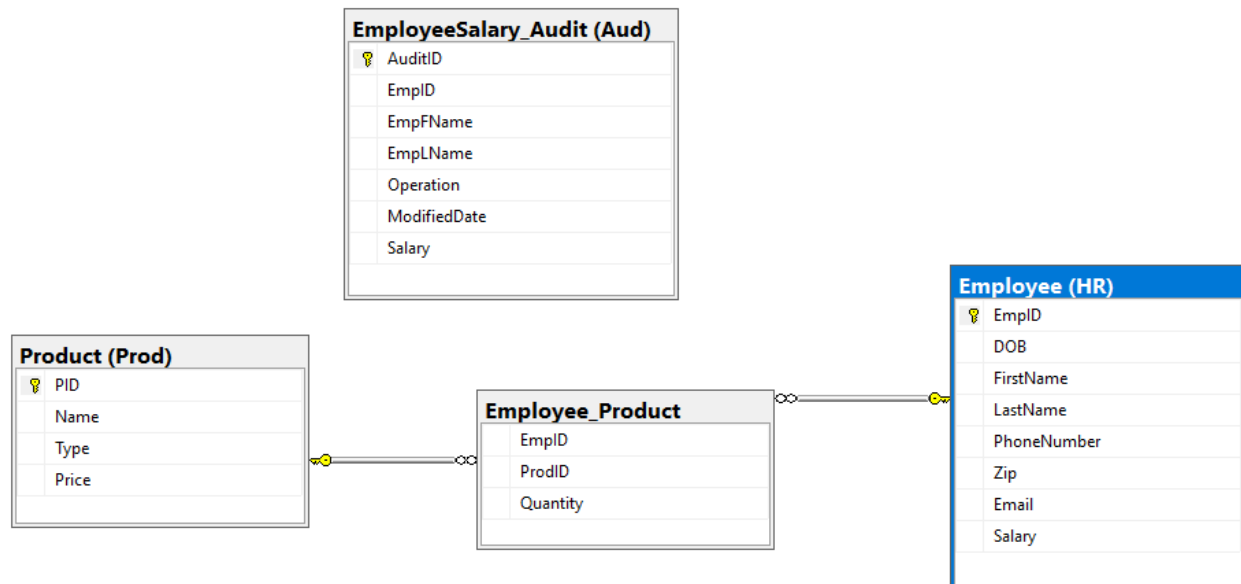
Table: **Aud.EmployeeSalary_Audit**

Column Name	Description	Data Type	Size	Identity	Unique	Default	Check	Allow Nulls	Index
AuditID	PK ; Unique Audit ID	smallint							Y
EmpFName	Employee First Name	varchar	40						
EmpLName	Employee Last Name	varchar	40						
Operation	Employee DOB	varchar	3						
ModifiedDate	Employee Phone #	date							
Salary	Employee Salary	money							

Table: **dbo.Employee_Product**

Column Name	Description	Data Type	Size	Identity	Unique	Default	Check	Allow Nulls	Index
EmpID	FK ; Employee ID	varchar	10						
ProdID	FK ; Product ID	smallint							
Quantity	Product Type	smallint							

Database diagram for each schema with PK/FK Relationships.



Checks and Defaults

The image displays three screenshots from SQL Server Enterprise Manager:

- Left Screenshot:** Object Explorer showing the 'Hospital' database. The 'Prod.Product' table is selected, and its 'Constraints' folder is expanded. The following constraints are highlighted in yellow:
 - CHK_DOB
 - CHK_Zipcode
 - DF_Employee_Zip_37A5467C
 - DF_Product_Type_3E52440B
- Middle Screenshot:** SQL Query window showing a connection to the 'Hospital' database.
- Right Screenshot:** Object Explorer showing the 'Hospital' database. The 'Defaults' folder is expanded, and the following defaults are highlighted in yellow:
 - dbo.typedefault
 - dbo.zipdefault

Rules

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Databases' folder is expanded, showing the 'Rules' folder under 'C_Ezema_Exam1'. Two rules are listed: 'dbo.EmpID_pattern_c' and 'dbo.Zip_pattern'. The right pane shows the execution of two SQL commands: `exec sp_bindrule EmpID_pattern_c, 'HR.Employee.EmpID'` and `exec sp_bindrule Zip_pattern, 'HR.Employee.Zip'`. The 'Messages' pane below shows the output: 'Rule bound to table column.' At the bottom, a status bar indicates 'Query executed successfully.' and the server name 'DESKTOP-UQCD3T1'.

Connect ▾

DESKTOP-UQCD3T1 (SQL Server 14.0.1000.169 - DESKTOP-UQCD3T1)

DESKTOP-UQCD3T1\MSSQLSERVER01 (SQL Server 14.0.1000.169)

DESKTOP-UQCD3T1\MSSQLSERVER02 (SQL Server 14.0.1000.169)

Databases

- System Databases
- Database Snapshots
- BullsOffice
- C_Ezema_Exam1
 - Database Diagrams
 - Tables
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Stored Procedures
 - Functions
 - Database Triggers
 - Assemblies
 - Types
 - Rules
 - dbo.EmpID_pattern_c
 - dbo.Zip_pattern
 - Defaults
 - Plan Guides
 - Sequences
- Service Broker
- Storage
- Security
- Hospital

```
exec sp_bindrule EmpID_pattern_c, 'HR.Employee.EmpID'
exec sp_bindrule Zip_pattern, 'HR.Employee.Zip'
```

100 % ▾

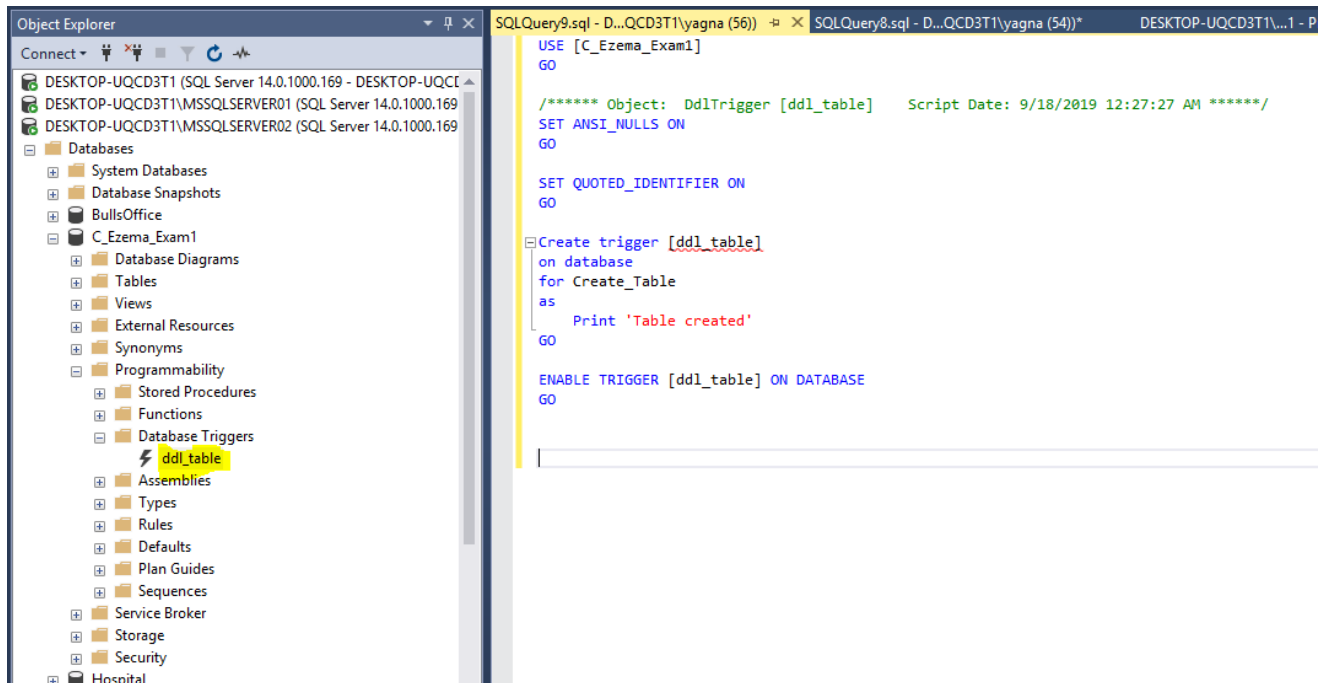
Messages

Rule bound to table column.

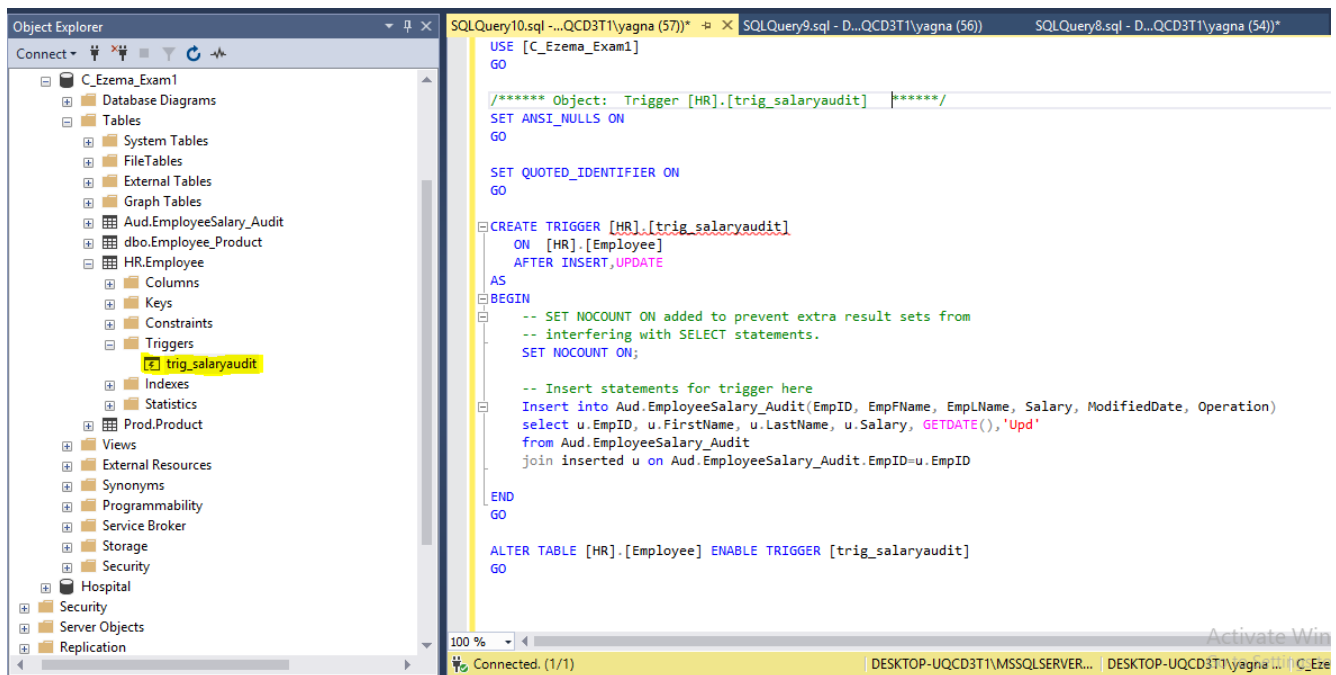
100 % ▾

✓ Query executed successfully. | DESKTOP-UQCD3T1

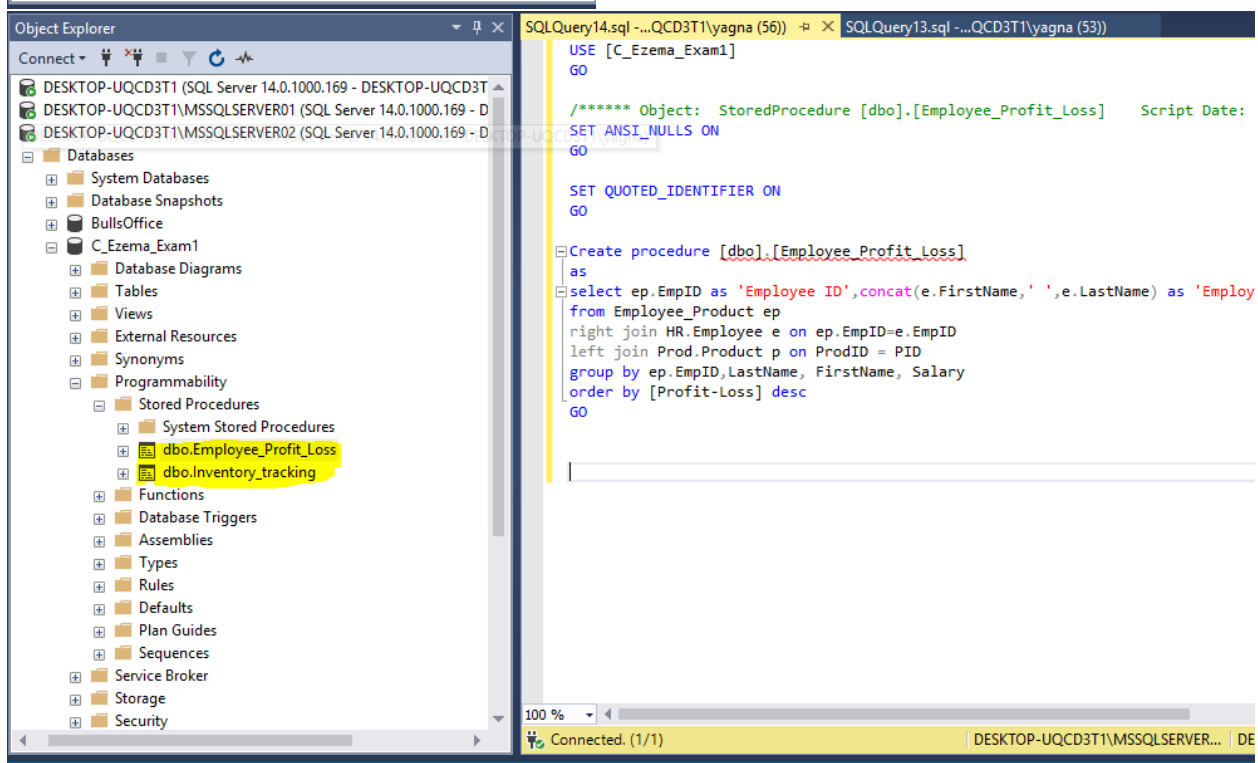
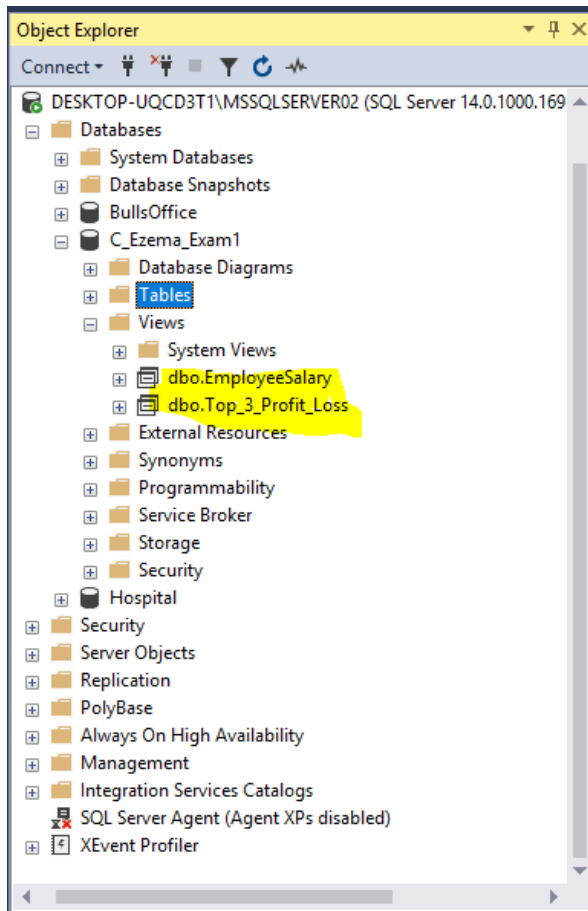
Triggers



Trigger 2



Views and SPROCs using Joins and Aggregators



```
USE [C_Ezema_Exam1]
GO

/***** Object: StoredProcedure [dbo].[Employee_Profit_Loss]    Script Date:
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

Create procedure [dbo].[Employee_Profit_Loss]
as
select ep.EmpID as 'Employee ID',concat(e.FirstName,' ',e.LastName) as 'Employ
from Employee_Product ep
right join HR.Employee e on ep.EmpID=e.EmpID
left join Prod.Product p on ProdID = PID
group by ep.EmpID,LastName, FirstName, Salary
order by [Profit-Loss] desc
GO
```

100 %

Connected. (1/1)

DESKTOP-UQCD3T1\MSSQLSERVER... | DE

USER STORIES

Left Join

As the store Manager James, I want to know my Top 5 employees so that they are recognized.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'C_Ezema_Exam1'. The main window shows a SQL query in the 'SQLQuery12.sql' file. The query is a SELECT statement that joins the 'Employee_Product' table with the 'HR.Employee' and 'Prod.Product' tables. It filters for the top 5 employees based on their profit-loss and orders them by profit-loss in descending order. The results are displayed in a table with 5 columns: Employee ID, Employeee, Salary, Revenue Generated, and Profit-Loss.

```
use C_Ezema_Exam1
--USER STORY
--AS store Mgr James, I want to compare employee's salary to revenue generated
select Top 5 ep.EmpID as 'Employee ID',concat(e.FirstName,' ',e.LastName) as 'Employeee', Salary
,ISNULL((sum(Quantity*Price)),0)as 'Revenue Generated', (ISNULL((sum(Quantity*Price)),0)-Salary)as 'Profit-Loss'
from Employee_Product ep
left join HR.Employee e on ep.EmpID=e.EmpID
left join Prod.Product p on ProdID = PID
group by ep.EmpID,LastName, FirstName, Salary
order by [Profit-Loss] desc
```

	Employee ID	Employeee	Salary	Revenue Generated	Profit-Loss
1	47-4777166	Sheeree O' Bee	46632.58	457016.48	410383.90
2	68-1219369	Donnie Itzkovich	52024.14	450026.33	398002.19
3	04-4309525	Mateo Rulton	71749.48	432415.43	360665.95
4	14-2298310	Caitrin Yeo	52466.60	359613.10	307146.50
5	42-7240346	Janice Schofield	94979.25	399917.85	304938.60

Logic Check:

Expanded the query results from showing only top 5 to showing profit-Loss for all employees

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'C_Ezema_Exam1'. The main window shows a SQL query in the 'SQLQuery12.sql' file. The query is a SELECT statement that joins the 'Employee_Product' table with the 'HR.Employee' and 'Prod.Product' tables. It filters for all employees based on their profit-loss and orders them by profit-loss in descending order. The results are displayed in a table with 5 columns: Employee ID, Employeee, Salary, Revenue Generated, and Profit-Loss.

```
use C_Ezema_Exam1
select ep.EmpID as 'Employee ID',concat(e.FirstName,' ',e.LastName) as 'Employeee', Salary
,ISNULL((sum(Quantity*Price)),0)as 'Revenue Generated', (ISNULL((sum(Quantity*Price)),0)-Salary)as 'Profit-Loss'
from Employee_Product ep
left join HR.Employee e on ep.EmpID=e.EmpID
left join Prod.Product p on ProdID = PID
group by ep.EmpID,LastName, FirstName, Salary
order by [Profit-Loss] desc
```

	Employee ID	Employeee	Salary	Revenue Generated	Profit-Loss
1	47-4777166	Sheeree O' Bee	46632.58	457016.48	410383.90
2	68-1219369	Donnie Itzkovich	52024.14	450026.33	398002.19
3	04-4309525	Mateo Rulton	71749.48	432415.43	360665.95
4	14-2298310	Caitrin Yeo	52466.60	359613.10	307146.50
5	42-7240346	Janice Schofield	94979.25	399917.85	304938.60
6	16-0590523	Carl Smalcombe	157088.47	454539.25	297450.78
7	31-6690004	Reidar Danielski	131808.15	423025.49	291217.34
8	73-3923686	Shanna Toplin	168169.93	426020.68	257850.75
9	87-5569283	Hayes Cocozza	183880.39	422561.74	238681.35
10	31-8008924	Bernardine Heskey	224685.32	456829.27	232143.95
11	33-8713341	Hadrian Ricciardelli	157109.01	385112.28	228003.27
12	78-6401480	Eleen Frankom	191169.68	412030.39	220860.71
13	42-9849406	Herb Rigts	248840.60	461498.07	212657.47
14	21-2528919	Megan Tumbridge	274642.38	472753.03	198110.65
15	38-2510974	Reiko Ambros	224165.07	389779.84	165614.77
16	00-3706539	Stacy Grigorushkin	179995.81	332551.67	152555.86

Full Join

As the sales manager Joan, I want to see all employees that have yet to sell any product so that they can be re-trained

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-UQCD3T1\MSSQLSERVER02'. The main window shows a query editor with the following SQL code:

```
--As the sales manager Joan, I want to see all employees that have yet to sell any product so that they can be re-trained
select isNull(e.EmpID, 'No ID') as 'Employee ID', concat(e.FirstName, ' ', e.LastName) as 'Employeee', Salary, Quantity
from HR.Employee e
Full outer join Employee_Product ep on e.EmpID=ep.EmpID
order by Quantity asc
```

The Results pane shows the output of the query:

Employee ID	Employeee	Salary	Quantity
59-1320849	Fina Jouannot	77269.03	NULL
81-6666904	Shoshana Otsmann	62759.71	NULL
87-5569283	Hayes Cocozza	183880.39	1
87-5569283	Hayes Cocozza	183880.39	1
87-5569283	Hayes Cocozza	183880.39	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
68-1219369	Donnie Itzkovich	52024.14	1
68-1219369	Donnie Itzkovich	52024.14	1
73-3923686	Shanna Toplin	168169.93	1
73-3923686	Shanna Toplin	168169.93	1
73-3923686	Shanna Toplin	168169.93	1

The status bar at the bottom indicates 'Query executed successfully.' and '181 rows'.

Logic Check:

Added a new user Johnny Test to see if he would be populated when query was refreshed

The screenshot shows the SQL Server Enterprise Manager interface. The query editor now includes an insert statement for a new user:

```
--Logic Check
insert into HR.Employee (EmpID, FirstName, LastName, DOB, PhoneNumber, Salary, Email, Zip) values ('04-4309815', 'Test', 'Johnny', '9/21/1997',
select isNull(e.EmpID, 'No ID') as 'Employee ID', concat(e.FirstName, ' ', e.LastName) as 'Employeee', Salary, Quantity
from HR.Employee e
Full outer join Employee_Product ep on e.EmpID=ep.EmpID
order by Quantity asc
```

The Results pane shows the output of the query, including the new user:

Employee ID	Employeee	Salary	Quantity
04-4309815	Test Johnny	51749.48	NULL
59-1320849	Fina Jouannot	77269.03	NULL
81-6666904	Shoshana Otsmann	62759.71	NULL
87-5569283	Hayes Cocozza	183880.39	1
87-5569283	Hayes Cocozza	183880.39	1
87-5569283	Hayes Cocozza	183880.39	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
78-6401480	Eleen Frankom	191169.68	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1
97-4495371	Cesar Hebbome	270446.28	1

The status bar at the bottom indicates 'Query executed successfully.' and '181 rows'.

Right Join

As store owner Jackie, I want to know the average earnings of each my top 3 sales rep so that they can used for benchmarking

The screenshot shows a SQL query in SQL Server Enterprise Manager. The query is a right join between Employee_Product and HR.Employee, with a left join to Prod.Product. It calculates the average commission earned for each employee, sorted in descending order, and returns the top 3 results.

```
select top 3 ep.EmpID as 'Employee ID',concat(e.FirstName,' ',e.LastName) as 'Employeee', Salary
,cast(ISNULL((avg(Quantity*Price)*.23),0)as numeric(18,2))as 'Avg Commission Earned'
from Employee_Product ep
right join HR.Employee e on ep.EmpID=e.EmpID
left join Prod.Product p on ProdID = PID
group by ep.EmpID,LastName, FirstName, Salary
order by [Avg Commission Earned] desc
```

	Employee ID	Employeee	Salary	Avg Commission Earned
1	31-8008924	Bernardine Heskey	224685.32	11674.53
2	21-2528919	Megan Tumbridge	274642.38	10873.32
3	42-9849406	Herb Rigts	248840.60	10614.46

Query executed successfully.

Logic Check :

Expanding from top 3 to see the avg earnings of all employees

The screenshot shows the same SQL query as above, but without the 'top 3' clause. The results table now displays the average commission earned for all 16 employees, sorted in descending order.

```
select ep.EmpID as 'Employee ID',concat(e.FirstName,' ',e.LastName) as 'Employeee', Salary
,cast(ISNULL((avg(Quantity*Price)*.23),0)as numeric(18,2))as 'Avg Commission Earned'
from Employee_Product ep
right join HR.Employee e on ep.EmpID=e.EmpID
left join Prod.Product p on ProdID = PID
group by ep.EmpID,LastName, FirstName, Salary
order by [Avg Commission Earned] desc
```

	Employee ID	Employeee	Salary	Avg Commission Earned
1	31-8008924	Bernardine Heskey	224685.32	11674.53
2	21-2528919	Megan Tumbridge	274642.38	10873.32
3	42-9849406	Herb Rigts	248840.60	10614.46
4	47-4777166	Sheeree O' Bee	46632.58	10511.38
5	16-0590523	Carl Smalcombe	157088.47	10454.40
6	68-1219369	Donnie Itzkovich	52024.14	10350.61
7	04-4309525	Mateo Rulton	71749.48	9945.55
8	73-3923686	Shanna Toplin	168169.93	9798.48
9	31-6690004	Reidar Danielski	131808.15	9729.59
10	87-5569283	Hayes Cocozza	183880.39	9718.92
11	78-6401480	Eleen Frankom	191169.68	9476.70
12	42-7240346	Janice Schofield	94979.25	9198.11
13	38-2510974	Reiko Ambrois	224165.07	8964.94
14	33-8713341	Hadrian Ricciardelli	157109.01	8857.58
15	35-5516328	Tony Benoist	227863.25	8728.28
16	14-2298310	Caitrin Yeo	52466.60	8271.10

Query executed successfully.